VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

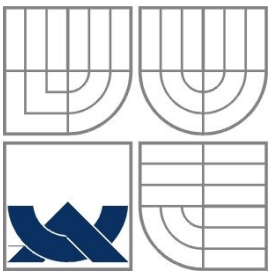
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

KOMUNIKACE SE ZAŘÍZENÍM TYPU USB HID

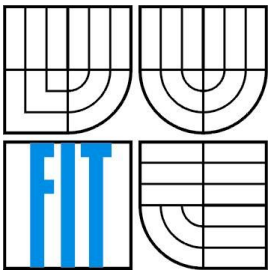
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE Jan Tylich
AUTHOR

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

KOMUNIKACE SE ZAŘÍZENÍM TYPU USB HID

USB HID COMMUNICATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Jan Tylich
AUTHOR

VEDOUCÍ PRÁCE — Ing. Zdeněk Vašíček
SUPERVISOR

BRNO 2010

Abstrakt

Tato práce se zabývá praktickou realizací propojení výukové platformy FITkit s modulem VDIP1 obsahujícího USB čip Vinculum firmy FTDI a návrhem knihovny umožňující komunikaci s USB zařízeními. Výstupem práce jsou dvě aplikace, které rozšiřují stávající možnosti komunikace kitu s člověkem přidáním počítačové klávesnice a myši.

Abstract

This thesis deals with a practical realisation of connection development module VDIP1 and education platform FITkit. Implementation of demonstration applications for communicating with USB HID devices connected from module VDIP1. These applications were designed for FITKit, an educational platform, to improve the existing user interface by incorporating keyboard and mouse.

Klíčová slova

FITkit, HID, klávesnice, myš, USB, Vinculum, VNC1L

Keywords

FITkit, HID, keyboard, mouse, USB, Vinculum, VNC1L

Citace

Jan Tylich: Komunikace se zařízením typu USB HID, bakalářská práce, Brno, FIT VUT v Brně, 2010

Komunikace se zařízením typu USB HID

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Zdeňka Vašíčka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jan Tylich
19. května 2010

Poděkování

Tímto děkuji vedoucímu mé práce Ing. Zdeňku Vašíčkovi za odborné vedení a čas věnovaný konzultacím. Především děkuji za pomoc s platformou FITkit.

Obsah

Úvod

FITkit je výuková platforma, která je vyvíjena na Ústavu počítačových systémů (UPSY). Jejím hlavním účelem je usnadnění výuky hardwarově orientovaných předmětů. K této platformě je možno skrze výstupy připojit typově různorodé periferie. Avšak jedno z nejpoužívanějších rozhraní posledních let zde chybí (USB rozhraní). Universální sériová sběrnice označovaná USB se od svého uvedení na trh stala nedílnou součástí našich životů. Právě skrze tuto sběrnici by bylo možno připojit k FITkitu opravdu široké spektrum dalších zařízení.

Tato práce se zabývá návrhem a realizací programové obsluhy USB zařízení ze skupiny HID, tedy Human Interface Device, mezi které patří počítačová klávesnice a myš. Doposud byl FITkit schopen jen velice omezené interakce s člověkem skrze šestnácti tlačítkovou klávesnici. Avšak standardní mezinárodní klávesnice má 102 kláves a umožňuje tak reakci na podstatně širší skupinu signálů. Tato práce se zabývá tím jakým způsobem lze pracovat s takovým rozšířením, včetně propojení s jednoduchým grafickým terminálem.

Jelikož platforma nemá USB sběrnici byl využit k projení s HID zařízením vývojový modul od britského výrobce FTDI. Konkrétně model VDIP1 osazeny programovatelnou součástkou umožňující vytvořit USB host port.

Práce je členěna následovně. První kapitola obsahuje stručný popis použitých komponent pro tuto práci (platformy FITkit, VDIP1 a klávesnice s myší). Dále rozebírá sběrnici USB a její využití, historii a specifika. Druhá a třetí kapitola popisuje implementaci projektu a zapojení celé sestavy. Nedílnou součástí je i popis sestavených funkcí usnadňující práci při dalším vývoji aplikací. Čtvrtá a pátá kapitola rozebírá praktické výsledky sestavených demonstračních aplikací a zapojení pro klávesnici a myš. Dále také komunikaci s dalšími periferiemi jako je LCD displej a monitor. Poslední kapitola nabízí nástin dalších možných rozšíření, vylepšení stávajících knihovních funkcí a návrhy potenciálně zajímavých zapojení.

1 Teoretický úvod

Jelikož komunikace s USB zařízeními je z pohledu FITkitu natolik komplikovaná záležitost, neobejdeme se zde bez specializovaného čipu. A protože se nejedná pouze o softwarový projekt je třeba nejdříve představit vybrané hardwarové prvky zapojení. Je zapotřebí tří základních komponent. Výukovou platformu FITkit, jež představuje srdce zapojení. Rozšiřující sběrnici USB v podobě vývojového modulu VDIP1 a USB periferní zařízení.

Nedílnou součástí teoretického úvodu je i představení struktury komunikačního protokolu USB HID zařízení a jeho provázanosti s modulem VDIP1.

1.1 Výuková platforma FITkit

Výuková platforma FITkit je určena k získávání praktických zkušeností s vývojem vestavěných systémů (anglicky Embedded Systems), jež jsou typickým příkladem využití informatiky v praxi. Takováto zařízení se nacházejí v našem bezprostředním okolí, kde často nahrazují specializovaným hardwarovým vybavením celý počítač. Mezi nejběžnější příklady patří mobilní telefony, MP3 přehrávače, dálková ovládání, krokoměry, kardiostimulátory a mnoho dalších.

V současné době se FITkit dostal již k třetí revizi označované jako FITkit 2.0. Pro práci byl vybrán právě tento model, proto se budou v následujícím textu vyskytovat informace výhradně k tomuto modelu. Pro zájemce o další informace a odlišnosti mezi verzemi je zde web projektu [1]. Kit je osazen nízkopříkonovým 16-bitovým RISC mikrokontrolérem (dále MCU z angl. Micro-Controller Unit) od firmy Texas Instruments. Přesněji model MSP430F2617 disponující 92kB FLASH a 8kB RAM. Důležitou součástí tohoto procesoru je sériové komunikační rozhraní UART, které je využíváno pro komunikaci s platformou Vinculum. Vývoj software pro MCU se tvoří v jazyce C a k překladu slouží GNU překladač MSP430-GCC.

Dále je na platformě programovatelné hradlové pole FPGA (Field-Programmable Gate Array) XC3S50-4PQ208C řady Spartan 3 firmy Xilinx. Struktura obvodů FPGA vzniká na základě popisu programovacím jazykem VHDL. Tento obvod je využit v části s grafickým textovým terminálem. Jedním z důležitých prvků pro tuto práci je USB převodník FT2232 firmy FTDI sloužící k napájení celého zařízení a komunikaci s počítačem. Poskytuje dva na sobě nezávislé kanály.

Jak již bylo zmíněno v úvodu práce, platforma je vybavena velkým množstvím portů pro připojení několika typů zařízení. Proto na desce naleznete VGA výstup, který slouží k připojení k monitoru. Audio rozhraní spolu s dvěma PS2 porty, dvouřádkový displej, klávesnici 4x4, RS232 a rozšiřující sloty to vše rozšiřitelné o ethernetový nebo bezdrátový síťový modul.

1.2 Vinculum VDIP1

Vývojový modul Vinculum VDIP1 slouží jako programovatelný USB Host/Slave řadič v různých aplikacích, a tak snižuje náklady a čas spojené s vývojem zařízení. O vývoj a výrobu se stará firma FTDI, která v roce 2006 uvedla na trh vestavěný USB 2.0 host řadič s označením VNC1L-1A. Tento obvod se stal základem pro celou řadu nových výrobků.

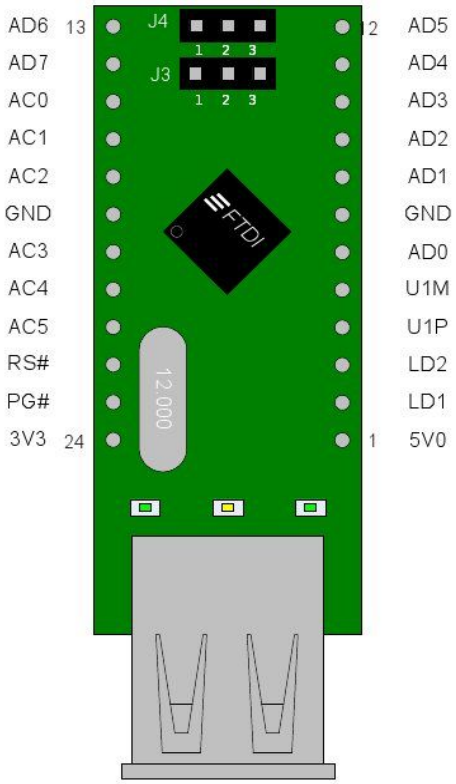
Obvod VNC1L obsahuje 2 USB Porty které mohou být firmwarem samostatně zvoleny jako host nebo slave porty. Modul VDIP1 je osazen pouze jedním USB portem, ale druhý je možno vyvést na nepájivé pole nebo desku plošných spojů. Modul například umožňuje spojení USB flash disků s mikrokontroléry přes Vinculum UART, SPI nebo FIFO rozhraní či propojení USB zařízení různých tříd jako Mass Storage Class, Printer Class a HID Class zařízení.

Obvod lze díky svým možnostem lehce využít pro propojení nejrůznějších zařízení a flash disku. Příkladem takového zapojení je propojení digitálního fotoaparátu s flash diskem, mobilního telefonu a flash disku, PDA zařízení spolu s flash diskem, dvou flash disků nebo dvou mp3 přehrávačů. Ovšem toto jsou zapojení která lze realizovat za pomocí firmwaru dodávaného výrobcem. Daleko širší možnosti skýtá vývoj vlastního firmware. Firma poskytla program a příručku, v které jsou vysvětleny možnosti a postupy při úpravách existujících firmware viz kapitola 1.2.2. Na stránkách výrobce [12] lze proto stáhnout nejen tuto příručku, ale i schéma obvodu a několik vzorových aplikací s podrobnou dokumentací. Další alternativou je propojení modulu s dalším mikrokontrolérem, který bude pomocí existujících příkazů firmwaru komunikovat s připojeným zařízením a případně dál zpracovávat získaná data. Příkazy SSU, DRD a DSD vlastně poskytují jakousi abstrakci nad USB HID Request viz kapitola 1.4.2. Tyto funkce dokáží odeslat požadavek a zpracovat i reakci. Například funkce DRD může zachytávat HID Report, jejím výstupem je čistá datová informace bez podružných deskriptorů.

Modul má poměrně malé rozměry a je navržen tak, aby jej bylo možné použít v nepájivém kontaktním poli. Širší rozměr desky včetně přečnívajícího USB portu má 51,63 mm. Užší strana desky plošného spoje čítá 18.10 mm a na výšku ční nad spojem do výše 7 mm USB port. Na vrchní straně je umístěno 6 pinů pro nastavení způsobu komunikace s modulem. Výběr probíhá pomocí propojek J3 a J4. Je zapotřebí dodat že lze pro I/O komunikaci použít jen jeden ze tří podporovaných rozhraní a volbu po přivedení napájení již nelze za běhu změnit.

Obvod má čtyřiadvacet pinů (viz obrázek číslo 1.1) z nichž některé mají pevně danou funkci jako je pin číslo 22 RS#, který slouží pro restartování obvodu z popudu externího zařízení. Například při nahrávání novějšího firmware. Dále je zde několik pinů pro napájení (1, 24, 7, 18) a výstupy LED

diod (2, 3). Většina ostatních pinů má funkci přiřazenou vždy podle vybraného komunikačního rozhraní viz katalogový list [2].



Obrázek 1.1: Modul VDIP1, převzato z katalogového listu [2]

1.2.1 Základní parametry obvodu

- ► 8/32 bitové V-MCU jádro
- ► Duální DMA kontroléry pro hardwarovou akceleraci (slouží k výměně dat mezi periferními obvody a pamětmi)
- ► 64k vestavěná flash paměť programu
- ► 4k interní Datová SRAM
- ► 2 x USB 2.0 Slow/Full speed host/slave porty
- ► UART, SPI a paralelní FIFO rozhraní
- ► PS2 rozhraní pro myš a klávesnici
- ► Až 28 GPIO (General purpose IO pins) pinů v závislosti na konfiguraci
- ► 3.3V operace s bezpečnými vstupy do 5V
- ► Nízká spotřeba (25mA pracovní režim/2mA klidový stav)
- ► Snadný update FTDI firmwaru
- ► LQFP-48 RoHS pouzdro
- ► Schopnost víceprocesorové konfigurace

1.2.2 Firmware

Výrobce připravil několik různých firmware pro specifická použití zde uvádím jejich přehled vycházející z uživatelské příručky [3]:

- **VDAP firmware (disk a periferní zařízení)**
  Původní firmware se kterým je modul zakoupen. Podporuje flash disky jen na portu 2, ostatní zařízení jako obvody FTDI, HID zařízení a další zvládají komunikaci na obou portech. Proto je vhodný především pro monitorovací činnost (ukládání log souborů na připojený disk).
- **VMSC firmware (hudební přehrávač)**
  Přináší podporu přehrávání hudby z přenosných disků. Podpora formátu MP3 a WMA do přenosu 320 kbps. Je vytvořena podpora pro základní příkazy k přehrávání, práce s ID3 tagy, skoky ve skladbě, práce s hlasitostí a pár dalších možností.
- **VDPS (disk, PC monitor a slave port)**
  VDPS nastavuje port USB2 jako master a USB1 jako slave, takže je možné s ním připojit modul k PC. U PC se následně chová jako běžný FTDI obvod, podporuje D2xx i VCP ovladače.
- **VCDC (zařízení typu Communication Class Device)**
  Práce s modemy, komunikace s telefony a ethernetové přípojky. Tento firmware poskytuje výrobce ke stažení na vyžádání.
- **VDIF firmware (disk a FTDI obvod)**
  Velice podobný firmware jako VDAP avšak má rozšířenou podporu obvodů FTDI. Při připojení FTDI obvodu do portu 1 se automaticky nastaví shodná přenosová rychlost a další přenosové parametry s obvodem VNC1L. Typicky se používá s obvody FT232B, FT232R nebo FT2232. Tento firmware je u výrobce ke stažení na vyžádání.
- **VDFC firmware (disky a kopírování)**
  Určen výhradně pro práci se dvěma USB flash disky, takže jsou implementovány především funkce pro práci se soubory jako je kopírování z disku na disk a vytváření záloh.

1.2.3 Rozhraní VNC1L

Obvod obsahuje tři rozhraní. Rozhraní UART, SPI a FIFO. Tento stručný popis je převzat a upraven [4], více informací lze dohledat v literatuře nebo na internetu.

UART

- klasický sériový port RS232, jako z PC, s plným osazením (8 pinů)
- pro běžnou komunikaci jsou potřeba piny RxD, TxD a bezpodmínečně i RTS a CTS
- LVTTL úrovně 3,3V, avšak 5V tolerantní, takže ho lze připojit přímo k procesoru na 5V
- přenosová rychlost je nastavitelná od 300 do 3000000 baudů
- výchozí nastavení: přenosová rychlost 9600 baud, 8 data bitů, 1 start bit, 1 stop bit, bez parity

SPI

- sériová externí sběrnice vhodná i k propojení více zařízení
- synchronní obousměrný přenos dat

FIFO

- klasická paralelní FIFO sběrnice
- obsahuje piny D0 – D7, -RD, WR, -TXE a -RXF

1.2.4 Režimy příkazů VNC1L

Obvod komunikuje s dalšími zařízeními nebo člověkem pomocí dvou různých vnitřních protokolů, mezi kterými je možno kdykoliv přepnout. Prvním z nich je výchozí Extended Command Set (ECS), který používá textové příkazy složené ze dvou až tří písmen vycházejících převážně z anglických zkratek činností které mají příkazy vykonávat. Příkladem je sada příkazů pro práci s flash diskem. Příkazy DIR a CD se chovají stejně jako je tomu v operačních systémech. Konkrétní sada příkazů se samozřejmě liší podle nahraného firmware, ale vždy je možné příkazy spustit z obou režimů. ECS je vhodný pro seznámení se s možnostmi obvodu a pro přímou komunikaci s člověkem.

Pro větší pohodlí při práci s obvodem je navíc možno v režimu ECS zvolit formát číselných odpovědí obvodu. Odpověď je zde myšlena jako reakce na příkaz. Opět je na výběr ze dvou možností a sice ASCII forma číselných odpovědí (IPA) nebo hexadecimální odpovědi (IPH). Zde bych zmínil, že výchozí režim při zapnutí je IPH a skutečnost, že si obvod pamatuje při přepnutí z ECS do SCS a zpět předešlou volbu. Přepnutím na IPA se z upovídaného protokolu stává ještě upovídanější a délka odpovědi je tak často oproti SCS více jak dvojnásobná.

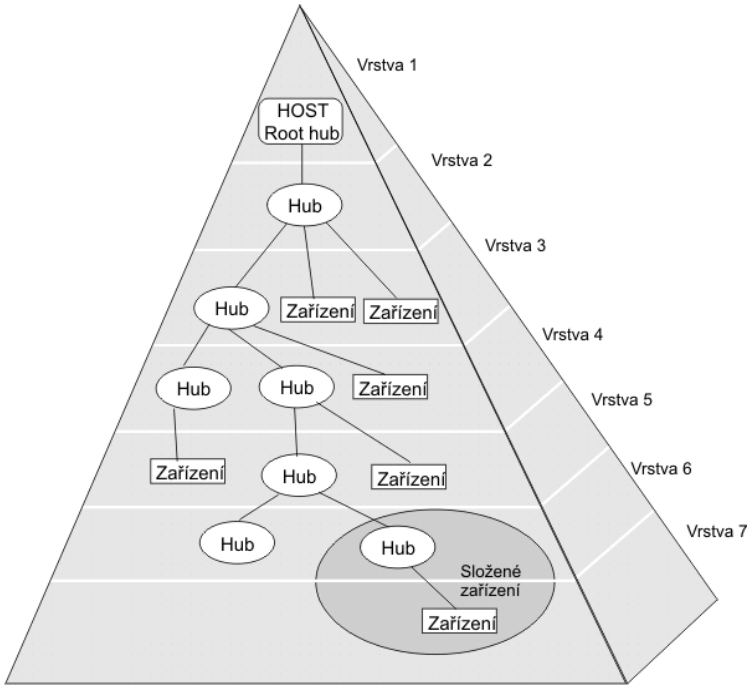
Druhým protokolem je Shortened Command Set (SCS) vhodný především pro programovou obsluhu obvodu a práci s připojenými zařízeními, neboť používá binární protokol. Skládá se z příkazů délky jednoho až dvou bytů a hexadecimální hodnoty parametru.

1.3 Sběrnice USB

„Universal Serial Bus (USB) je komunikační architektura, která poskytuje osobnímu počítači (PC) možnost propojení různých zařízení pomocí jednoduchého čtyř-žilného kabelu. USB je vlastně dvoudrátovou sériovou komunikační linkou, která běží buď na 1,5 nebo 12 megabitech za sekundu (Mbit/s). USB protokoly lze nakonfigurovat zařízení při startu, nebo až když jsou připojeny za běhu. Tato zařízení jsou rozdělena do různých tříd. Každá třída definuje společné chování a protokoly pro zařízení, která slouží k podobným funkcím. [5]“

V současné době je nejnovější specifikací USB 3.0, která se odklání od původního čtyř-žilného kabelu a rozšiřuje ho o další čtyři vodiče. Jednotlivé USB sběrnice i s konektory jsou zpětně kompatibilní. Přenosové rychlosti se postupně navyšovaly proto dnes existují čtyři rychlostní standarty. USB 1.0 a 1.1 podporují pouze režimy Low speed - 1.5 Mbit/s a Full speed - 12 Mbit/s. USB 2.0 přidává High speed s teoretickou rychlostí až 480 Mbit/s a konečně nejnovější USB 3.0 navyšuje rychlost až k současnému teoretickému maximu 5 Gbit/s v režimu Super Speed.

Do sběrnice lze připojit až 127 zařízení. Maximální délka kabelu mezi dvěma zařízeními je 5 metrů. Zařízení jsou zapojena v hvězdicové topologii, tak že další rozšiřující porty poskytují USB huby. Huby jsou zároveň středy hvězdic a někdy mají vlastní napájení, tak aby pokryly potřeby připojených zařízení. Každá sběrnice má jeden takzvaný kořenový rozbočovač (root hub) do kterého se připojují další zařízení a huby. Celá struktura zapojení je s ohledem na zpoždění vodičů maximálně sedmivrstvá a znázorňuje ji obrázek číslo 1.2.



Obrázek 1.2: Pyramidová struktura USB, převzato z [6]

USB sběrnice může zařízením zároveň poskytnout i napájecí napětí v rozsahu 5 a 0,25 V. Pokud zařízení stačí proud 100mA může být napájeno pouze skrze USB port a nemusí ani žádat o napájení (limit je 500mA). Takové množství energie postačuje flash diskům, počítačovým myším, klávesnicím nebo různým hračkám do USB portu. Avšak zařízení jako 3.5" externí disk potřebují víc proudu, proto mívají svůj vlastní zdroj napájení.

1.4 Zařízení typu HID

Human interface device (HID) zařízení přímo kooperují s člověkem a převádí tak podměty od člověka k počítači a zpět. Mezi typická HID zařízení patří klávesnice, myši (a její náhrady jako toutchpad, trackball, trackpoint), grafické tablety, joysticky, gamepady, webkamery, sluchátka s mikrofonem a nespočet dalších méně obvyklých zařízení. Jelikož jsou zařízení připojena přes USB chovají se jako Plug & Play zařízení, která jsou ihned po připojení schopna provádět svou činnost. Tedy zajistit si napájení počáteční konfiguraci a zahájit samotnou činnost.

Dokument *Device Class Definition for HID 1.11* [5] popisuje tuto třídu, definuje konfiguraci, komunikační protokol a standardizuje tak základní funkčnost těchto zařízení. Současně však ponechává volný prostor pro rozšíření komunikačního protokolu, která může zavést výrobce určité USB periferie. Znamená to že například u počítačové myši je předem vyhrazen byte pro detekci stisku tlačítek a jsou zde stanoveny tři hodnoty pro stisk tlačítek. Výrobce tak může využít zbylý prostor v bytu pro dalších pět tlačítek a přidat další byte pro identifikaci nastavení citlivosti. Takto rozšířená myš bez použití ovladačů bude fungovat, ale reagovat budou pouze tři tlačítka a teprve jakmile se nahraje ovladač od výrobce, zprovozní se rozšiřující funkce dalších tlačítek jako přepínání citlivosti nebo informace o stavu akumulátoru u bezdrátových myší.

Pro přesné pochopení jak k výměně těchto informací dochází je třeba si uvědomit, že každá z USB tříd se může skládat ještě z podtříd a k nim definovaným protokolů. Takto se sdružují zařízení s podobnými funkcemi a potřebami konfigurací. Toto podrobnější rozdělení se týká i třídy HID viz tabulka 1.1.

Třída	Podtřída	Protokol	Význam
03h	00h – žádná 01h – Boot Interface	00h	nespecifický
		01h	klávesnice
		02h	myš

Tabulka 1.1: Rozdělení třídy HID

1.4.1 Deskriptory (HID Class-Specific Descriptors)

Jediné USB zařízení může mít několik konfigurací, například pro Low speed a Full speed komunikaci. Proto je v ROM paměti zařízení uloženo nastavení v podobě takzvaných deskriptorů. HID třída používá takzvané class-specific descriptors, které se liší od obyčejných USB deskriptorů.

- **HID Descriptor**

  HID Descriptor určuje počet, typ a velikost zprávy HID deskriptorů a fyzikální deskriptory, které jsou spojeny se zařízením třídy HID.

- **HID Descriptor Report**

  Je vlastně zpráva o formátu dat, která mohou zařízení posílat nebo přijímat a přitom nejsou definována ve specifikaci třídy HID. Proto je zařízení HID povinno poskytnout HID Descriptor Report, která obsahuje všechna datová pole. jež. HID zařízení může používat. Pro každé pole v HID Descriptor Report je jeden deskriptor, který definuje kolik bitů konkrétní datová položka zabírá, jaké má využití, jaké je rozmezí hodnot, které používá, atd.

1.4.2 Požadavky (HID Class-Specific Requests)

O získání těchto informací lze požádat zasláním požadavků na stav, nastavení nebo přímo nastavit stav. Požadavky jsou následující:

- Získej zprávu (GET_REPORT)
- Nastav zprávu (SET_REPORT)
- Získej nečinnost (GET_IDLE)
- Nastav nečinnost (SET_IDLE)
- Získej protokol (GET_PROTOCOL)
- Nastav protokol (SET_PROTOCOL)

1.5 Klávesnice

Klávesnice je nedílnou součástí počítače již od jejich počátku. Vychází z klávesnic pro psací stroje, které byly paradoxně navrženy pro pomalejší psaní, aby nedocházelo k zachytávání typových pák. Standardní mezinárodní Windows klávesnice mají 105 kláves. Klávesnice se v dnešní době vyskytují v nejrůznějších podobách a rozloženích.

Rozložení QWERTZ je základem pro českou normu ČSN 36 9050 což je národní standard. Kromě tohoto rozložení se u nás používá rozložení QWERTY převzaté z anglicky mluvících zemí a využívané především programátory. Další rozložení jsou specifická pro jazyky a účely používání klávesnice, proto se používají i její redukované formy.

Připojení se v průběhu času měnilo od 5-pin konektorů DIN k jejich zmenšené verzi PS/2 až k dnešním USB a bezdrátovým klávesnicím.

1.6 Počítačová myš

„Počítačová myš je malé polohovací zařízení, které převádí informace o změně své pozice na povrchu plochy (např. desce stolu) do počítače, což se obvykle projevuje na monitoru jako pohyb kurzoru. Nachází se na ní jedno či více tlačítek, může obsahovat jedno i více koleček pro usnadnění pohybu v dokumentu. Ze spodní strany nalezneme zařízení snímající pohyb. [7]“

2 Návrh

Kapitola se zabývá dočasnými zapojeními, seznámením s funkcemi obvodů a výběrem firmware.

2.1 Seznámení s VDIP1 a výběr firmware

Jelikož bylo plánováno v budoucnu propojit MCU FITkitu a modul VDIP1 skrze sériovou linku bylo pomocí propojek J3 a J4 na modulu vybráno pro komunikaci sériové rozhraní. Katalogový list k modulu [2] dále uvádí na kterých pinech se lze propojit viz následující tabulka číslo 2.1.

Číslo pinu	Pojmenování	Typ přenosu	Popis
6	TxD	Výstup	Výstup pro odesílání asynchronních dat
8	RxD	Vstup	Vstup pro zachytávání asynchronních dat
9	RTS#	Výstup	Výzva k vysílání
10	CTS#	Vstup	Pohotovost k vysílání
11	DTR#	Výstup	Pohotovost koncového zařízení
12	DSR#	Vstup	Pohotovost ukončujícího zařízení
13	DCD#	Vstup	Detektor nosného signálu
14	RI#	Vstup	Příznak naplnění vyrovnávacího (přijímacího) registru platnými daty
15	TxDEN#	Vstup	Povolení vysílání RS485

Tabulka 2.1: Nastavení I/O pinů pro rozhraní UART. Přeloženo z katalogového listu [2]

I/O mode	J3	J4
UART	1-2	1-2
SPI	2-3	1-2
FIFO	1-2	2-3
UART	2-3	2-3

Tabulka 2.2: Piny J3 a J4 pro nastavení rozhraní. Převzata z [4] a odkazuje na obrázek 1.1 a 3.2

Pro první seznámení s modulem a ověření činnosti bylo zvoleno zapojení, které používá čip FTDI FT2232D pro převod sériové linky na USB. Bylo tak dosaženo zapojení USB->UART->USB. K navázání spojení postačí TxD (6) spolu s RxD (8). Vstup CTS# (10) je zapotřebí uzemnit jinak bude docházet k opakovaným restartům modulu. Jelikož byl prováděn vývoj v prostředí Linuxu nebylo zapotřebí žádných dodatečných ovladačů. V systému se vytvořil virtuální USB port a pomocí hyperterminálového klienta Putty se dalo s modulem pracovat.

První krůčky vedly k aktualizaci původního firmware na nejnovější verzi 2.68 (květen 2010). Postup je dobře popsán v manuálu k modulu [2] a samotná akce trvá zhruba 3 sekundy a je zcela automatická. Po připojení flash disku s patřičným souborem firmwaru se modul o vše postará sám.

Jak již bylo zmíněno výchozím firmwarem je VDAP, který má obecnou podporu pro disky na portu 2 a základní funkce pro práci s FTDI a HID zařízeními na obou portech. Modul VDIP1 má vyveden pouze port 2, a proto bylo možné vyzkoušet práci s diskem i HID zařízeními.

Práce s flash diskem

Po připojení disku dojde k jeho detekci a modul odpoví textem „Device detected P2“, za kterým následuje prohledání systému souborů (podpora systémů FAT). Tato akce trvá v závislosti na velikosti média od sekund až k minutám strávených čekáním na text „No Upgrade“. Po tomto informačním sdělení je zaslána pobídka „D:\>“ signalizující připravenost na další akce.

2.2 Připojení modulu k FITkitu

Předchozí způsob zapojeni využívající modul obsahující FT2232 byl vhodný pouze pro seznámení s obvodem a ověření jeho činnosti.

Uvnitř FITkitu je stejný obvod firmy FTDI řady FT2232, který opět poslouží ke komunikaci. Navíc použitím pinů s 3,3 V je dosaženo dostatečného napájecího potenciálu pro modul VDIP1. Přesné zapojení napájecích pinů je uvedeno v sekci Technické propojení 2.5, protože se shoduje s výsledným zapojením.

Obvod FT2232 má dva samostatné kanály. *„Kanál B je připojen k programovacím pinům mikrokontroléru (RESET, TST) a dále k pinům sériového rozhraní (RxD, TxD). Pomocí tohoto kanálu lze tedy mikrokontrolér programovat a komunikovat s ním přes terminálový program. [1]“* Kanál A byl použit pro komunikaci s modulem VDIP1. Tohoto stavu bylo dosaženo naprogramováním FPGA tak, že došlo k vytvoření propojení mezi kanálem A a dvěma volnými piny na jedné z 50-pinových patic FITkitu.

2.3 Potřebný software

Jelikož jde o práci spojenou s FITkitem nedalo se obejít bez patřičného softwarového vybavení. Protože vývoj probíhal pod systémem Ubuntu bylo možno využít stávajícího repozitáře software na školním serveru Merlin pro snadnou instalaci skriptovatelného terminálu QDevKit.

Pro následný překlad je zapotřebí dodat ještě překladač MSP430-GCC, taktéž dostupný jako balíček. A v neposlední řadě ISE WebPack nezbytný pro překlad zdrojových kódů psaných v jazyce VHDL. Poslední program vyžaduje vygenerování licence na jeho používání na stránkách výrobce.

2.4 Připojení Vinculum k PC

Jak bylo zmíněno v předchozích odstavcích této práce, tak než byly vytvořeny jednotlivé výsledné funkce popisované v pozdějších kapitolách došlo k sepsání jejich vzorů. Výstup ze sériového portu modulu byl skrze FITkit zpracován procesorem počítače.

Hlavní část práce proto proběhla v tomto zapojení. Při přepisu kódu do spustitelné podoby pro MCU došlo k pár změnám způsobených především nastavením překladače. A následně i výpisů do terminálové aplikace za pomocí knihoven *libfitkit*.

Hlavní rozdíl tohoto návrhu oproti výsledku spočívá v práci s čtením příchozích zpráv. V tomto řešení je použita funkce *read()* z knihovny *libkitchannel.h.* Tato funkce načítá příchozí znaky ze sériové linky až do dovršení určené hranice nebo dokud nevyprší předem stanovený časový limit (timeout). V případě že po lince nepřichází žádná data ukončí čekací smyčku. Oproti tomu funkce *read_rx()* vytvořená později načítá až do příchodu znaku <CR>. Tím by měl být ukončen veškerý obousměrný přenos s čipem Vinculum.

2.5 Informace o použitých HID zařízeních

Většina vývoje a testování demonstrační aplikace pro klávesnici byla prováděna se standardní klávesnicí Genius KB-200e. Jako zástupce multimediálních klávesnic byla použita Trust KB-2100E, která mimo jiné využívala i druhý kanál pro komunikaci. Kromě těchto klávesnic bylo otestováno i ještě pět dalších modelů. Aplikace vykazovala pro všechny klávesnice stejné a bezproblémové chování.

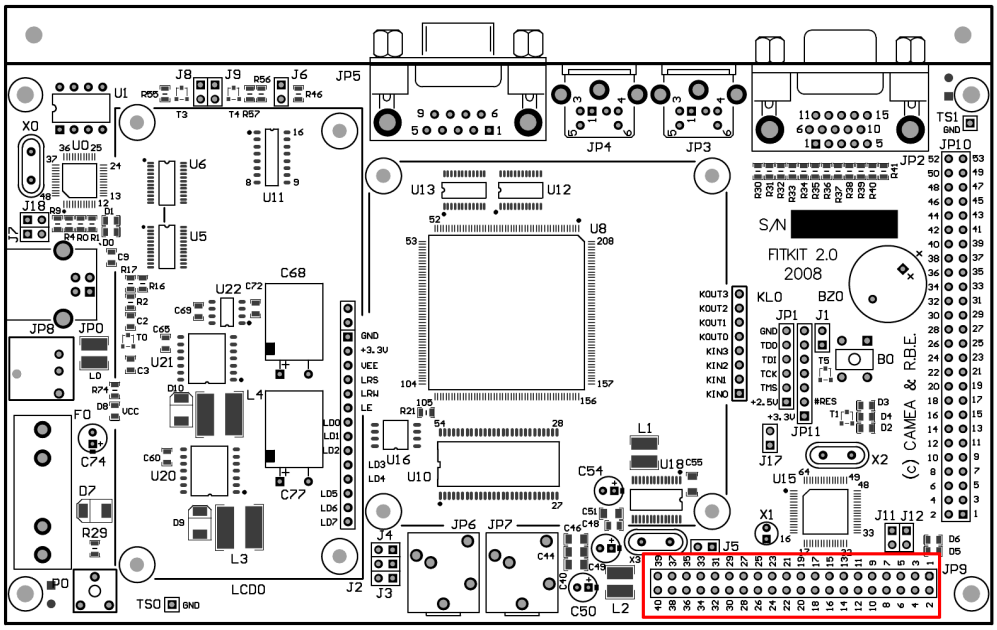
Jako opravdu funkční myš, která byla prožita pro vývoj aplikace se dá uvést Genius NetScroll 310. Pakliže myš dokáže ragovat na příkaz obvodu QP, bude vykonávat aplikaci správně. Kromě této myši byla testována i nejlevnější myš Genius NetScroll 110, ta však ihned po připojení doslova zamrazila obvod. Myš nezískala ani napájení a obvod kompletně přestal reagovat na příchozí komunikaci. Třetí testovanou myší je Genius NetScroll+ Mini Traveler, která získá napájení ihned po připojení bohužel obvod její přítomnost dál nedetekuje a nelze s ní tedy pracovat.

Kromě klávesnic a myší bylo k obvodu Vinculum připojeno i několik flash disků. Všechny disky se spolehlivě řídily příkazy zasílanými k obvodu Poslední připojená zařízení jsou huby. Otestován byl hub pro USB 1.1 i novější typ pro USB 2.0. Oba tyto huby poskytly ihned po připojení dalších zařízení napájení pro připojené zařízení, ale zatím se nepovedlo s nimi komunikovat i přesto, že samostatně reagují.

3 Implementace

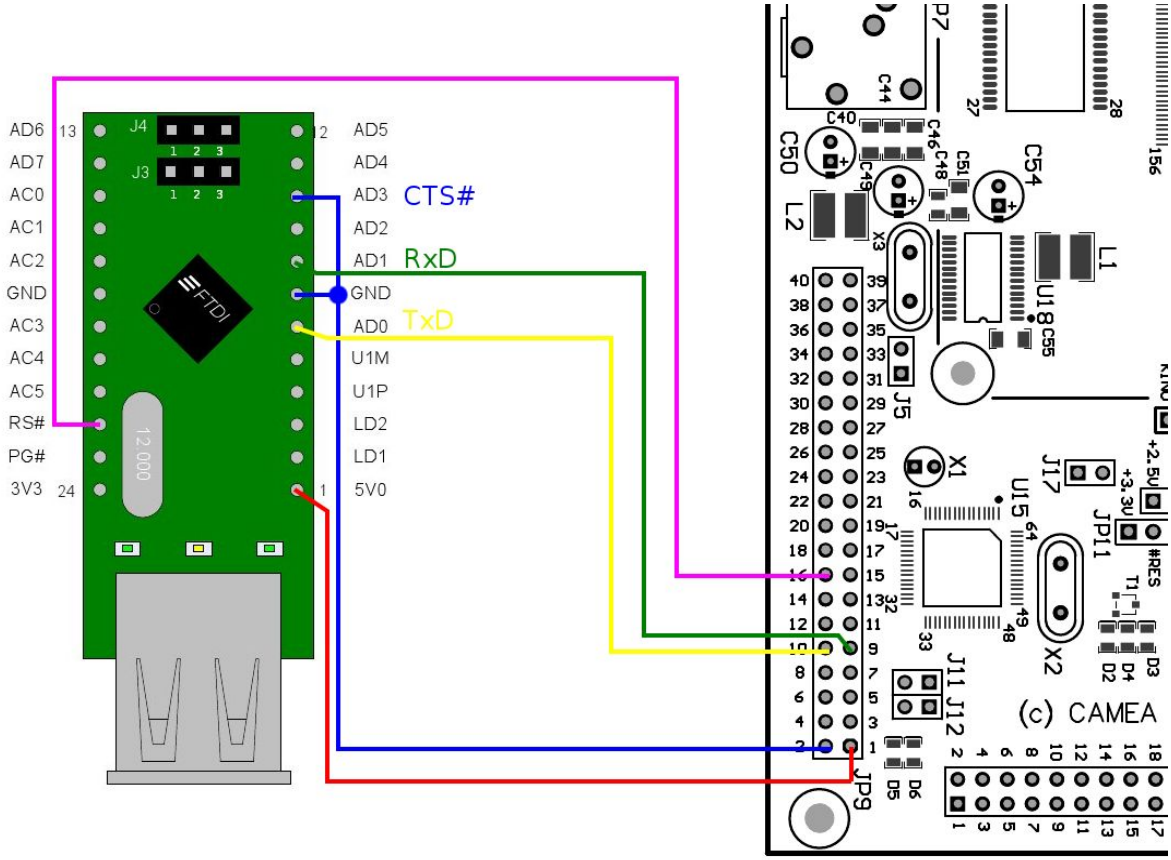
3.1 Technické propojení modulů

Pro výsledné propojení FITkitu a VDIP1 byla tedy použita sériová linka. O nastavení se starají piny J3 a J4 na modulu viz tabulka 2.2 v kapitole 2.1. Napájecí a všechny další potřebné piny jsou vyvedeny k FITkitu na jeho universální 50-pinové patici JP9 umístěné v levém spodním okraji desky. Poloha patice je znázorněna na obrázku 3.1 červeným obdélníkem.



Obrázek 3.1: FITkit otočený o 180° se zvýrazněnou paticí JP9, převzat a upraven z [1]

Přesný popis rozložení jednotlivých připojených pinů je uveden na dalším obrázku číslo 3.2. Zde je dobře vidět že pin 5V0 (1) na modulu VDIP1 je připojen na 1 pin patice JP9 a přivádí tak napětí z FITkitu. Aby bylo napájení kompletní je zapotřebí také propojit pin GND (7 nebo 18) s 2 pinem patice. Uzemnit je třeba i pin AD3 (10) na modulu představující CTS# rozhraní UART, jinak by docházelo k opakovaným restartům modulu Vinculum. Pin AD0 (6) představuje přijímací linku sériového rozhraní a je propojen s 10 pinem na patici. Pro vysílání je nezbytně nutné připojit ještě pin AD1 (8) na 9 pin patice. Posledním zbývajícím nepopsaným vodičem na obrázku je pin RS# (23) sloužící pro softwarový restart modulu VDIP1 a je vhodné ho spojit s 16 pinem na patici, jinak by nebyl možný restart. Takto propojené moduly lze společně řídit programem uvnitř MCU FITkitu.



Obrázek 3.2: Zapojení modulu VDIP1 a FITkitu

3.2 Popis knihovních funkcí a proměnných

3.2.1 Globální proměnné

Pro lepší přehlednost mezi předávanými argumenty funkcí bylo zvoleno několik globálních proměnných.

Hlavní proměnnou je buffer pro načítání přijatých zpráv po sériové lince s názvem **VinculumBuf.** Tento buffer má pevně stanovenou velikost danou preprocesorovou proměnnou BLEN nastavenou na výchozích 50 hodnot typu *char*. Do bufferu se tedy může uložit až 50 načtených znaků, avšak při práci s SCS by neměla být přijata zpráva přesahující 40 znaků. Díky tomuto řešení lze prakticky kdekoliv přistoupit k načtené zprávě a dále ji zpracovávat nebo jinak upravovat.

Další proměnnou je **DeviceChange**, která indikuje změnu ve stavu připojených zařízení od posledního čtení ze sériové linky. Nabývá pěti hodnot tak, aby signalizovala připojení nebo odpojení zařízení na obou portech.

Poslední dvě proměnné jsou zde, aby přinesly přehled do do toho, který z režimů příkazů čipu Vinculum je zrovna nastaven (více v kapitole 1.2.4). Pro tyto účely je zde boolovská proměnná **ECS**, která signalizuje který z režimů (ECS nebo SCS) je právě nastaven. Druhou boolovskou proměnnou je **IPH** poskytující informaci o režimu výpisu číselných hodnot v režimu ECS. Obě tyto hodnoty jsou po restartu modulu nastaveny na 1.

3.2.2 Pole informačních výstupů

V programu jsou definovány dvě textová pole reprezentující textové informace předávané z obvodu Vinculum. Jsou zde aby při práci v SCS bylo možné informovat obsluhu programu plnohodnotným textovým oznámením v terminálovém okně.

Prvním polem je **DeviceType**, které nese informace o typu právě připojeného zařízení. Zda jde o FTDI obvod, HID zařízení nebo tiskárnu. Více informací je v textu o funkci *qp()*.

Druhé pole s názvem **MS** má v sobě uchované všechny typy zpráv zasílaných obvodem. Vždy je zde zpráva v ECS i SCS. Lze tak při přijetí zprávy v režimu SCS vypsat do terminálu plnohodnotnou informační textovou zprávu zasílanou původně jen v režimu ECS. Významy těchto zpráv jsou vcelku jednoznačné, ale v případě nejasností lze v manuálu k firmware [3] nalézt podrobnější vysvětlení významu.

3.2.3 Načítání dat ze sériové linky

Funkce *read_rx()* zajišťuje bezproblémové načítání příchozích zpráv ze sériové linky. Vždy vynuluje vstupní globální buffer a poté načítá příchozí znaky. Pěkně znak po znaku a zapisuje je do bufferu VinculumBuf. Průběžně inkrementuje počítadlo načtených znaků a kontroluje jestli nedošlo ke konci přenosu signalizovaného příchozím znakem <CR>. Jako výstupní hodnotu vrací počet načtených znaků. Navíc se stará o výpis a signalizaci změn v připojených zařízení proměnnou DeviceChange.

Čekací smyčku ukončí příchozí znak <CR> nebo dosažení maximální kapacity bufferu.

3.2.4 Odeslání příkazů po sériové lince

O tuto část se stará velice jednoduchá funkce *write_tx()*, která vstupní řetězec znak po znaku odešle po lince do obvodu. Není zde žádná kontrola úspěšného přenosu. Ta probíhá až při načtení odpovědi. Parametry komunikace, jako je rychlost přenosu, počet stop bitů a parity jsou nastaveny preprocesorovou rutinou. Výchozí hodnoty sice lze změnit pomocí příkazů, na které obvod Vinculum reaguje, avšak došlo by k nekorektnímu chování celé aplikace.

V případě nutnosti změny by bylo zapotřebí spustit aplikaci s výchozím nastavením, odeslat žádost o změnu směrem k obvodu a následně parametr pozměnit i při čtení a odesílání. V současné době by to ale vyžadovalo změny ve funkcích obsluhující čtení a zápis na sériovou linku.

Výchozí hodnoty obvodu Vinculum jsou nastaveny na přenosovou rychlost 9600 baud, 8 datových bitů, 1 start bit a 1 stop bit bez parity se zapnutým RTS/CTS hardwarovým handshakingem.

3.2.5 Hardwarový restart

Na obrázku 2.2 je vidět jakým způsobem je propojen pin RS#, který zajišťuje hardwarovou obsluhu přerušení odesláním pulsu na příslušný pin. Vše je realizováno preprocesorovým makrem *VincRST()*.

3.2.6 Echo

Na kontrolu spojení je vytvořena funkce *sync()*, která použije dvojitý příkaz echo. Odešle se malé písmeno „e“ a posléze, když přijde odpověď v podobě stejného znaku se vše opakuje pro znak „E“. Při úspěšné reakci na dva po sobě jdoucí příkazy echo se smyčka ukončí a program může pokračovat dále ve svém běhu.

3.2.7 Funkce přepínající mezi komunikačními režimy

Jak již bylo popsáno v sekci „Režimy příkazů VNC1L“ existují dva základní režimy. Pro jejich přepínání existují funkce pojmenované shodně s jejich názvy. Obě funkce pracují s globální proměnnou ECS a nevyžadují žádné vstupní parametry. Taktéž jsou bez výstupu, protože je zajištěno jejich úspěšné provedení změny.

V zásadě jde o smyčku čekající na znak určující okamžik, kdy je možné vkládat příkazy (tzv. prompt) ve správném tvaru. Proto funkce *scs()* vyčkává na příchod zprávy „>“ naproti tomu *ecs()* musí zachytit zprávu „D:\>“. Obě samozřejmě ukončeny hodnotou <CR>.V případě že požadovaný režim již je aktivní nedojde k žádným přenosům mezi MCU a obvodem Vinculum.

Zbylé dvě funkce pro přepínání režimů číselných odpovědí lze samozřejmě spustit jen pokud je zvolen režim ECS. Funkce *ipa()* i *iph()* se v podstatě chovají dosti obdobně jako předchozí dvě funkce s tím rozdílem, že navracejí boolovskou hodnotu po svém ukončení. Globální proměnná IPH indikuje, momentální stav režimu.

3.2.8 Identifikace připojeného zařízení

Jak již bylo zmíněno, o samotnou detekci změny připojení nebo odpojení se stará funkce *read_rx()* změnou proměnné DeviceChange. Avšak funkce s názvem *qp()* dokáže při svém zavolání vypsat do terminálové aplikace typ připojeného zařízení. Funkce má jeden povinný parametr a to číslo portu. V případě modulu VDIP1 bez vyvedeného externího portu se tedy bude vždy volat s hodnotou 2. Funkce je navržena tak, aby byla spustitelná jen v režimu SCS.

Ve svém středu využívá funkci *DeviceIdent()*, která zajišťuje samotný výpis do terminálu. Jde o funkci pracující s textovým polem **DeviceType**. Tato funkce pomocí bitových posunů rozpozná, který bit je aktivní a podle jeho váhy přiřadí patřičnou zprávu pro výpis.

Do budoucna by možná bylo vhodné zařadit funkci *qp()* ihned po rozpoznání připojení nového zařízení v rámci podmínky uvnitř funkce *read_rx()*. Avšak toto řešení by nebylo zcela spolehlivé, protože některá zařízení nekomunikují zcela korektně či vůbec odmítají spolupracovat.

Pro přesnější identifikaci připojeného zařízení lze využít ještě dvou dalších příkazů obvodu Vinculum. Prvním je příkaz QD, který poskytne 32 byte dlouhou informaci, z které lze vyčíst VID a PID připojeného zařízení. Dále třídu a podtřídu, podporované rychlosti a několik dalších věcí. Bohužel tento příkaz při programování ne ve všech případech vracel očekávané informace. Zdá se proto, že některá zařízení sice obvod detekuje, ale to ještě není záruka bezproblémové obsluhy. Teoreticky by bylo možno z vyčtených informací zjistit zda byla připojena klávesnice nebo myš. Tento příkaz byl v době dočasného návrhu přepsán do funkce, která zajišťovala rozčlenění výsledků do srozumitelné formy. Avšak pro svoji nespolehlivost byl odstraněn.

Druhý příkaz se zaměřuje na flash disky. Jmenuje se IDD a kromě VID a PID vrací i velikost sektoru nebo informace o souborovém systému a volném a využitém místě.

3.2.9 Nastavení komunikačního kanálu

Pro přímý přístup k připojeným zařízením je zapotřebí zvolit si kanál pro komunikaci. Tato volba je nezbytně nutná pro provádění příkazů obvodu jako je zachycení dat od zařízení (DRD), zaslání dat zařízení (DSD) a zaslání nastavovacích dat (SSU).

K tomu slouží funkce *sc()*, která má jeden povinný parametr. Číslo v rozsahu 0 až 15, které udává na kterém kanále je možno se zařízením komunikovat. Nejčastější volnou je číslo 0. Mezi kanály lze kdykoliv přepínat. Avšak v případě připojení druhého zařízení na druhém portu je třeba si uvědomit, že kanál 0 získá dříve připojené a rozpoznané zařízení. Navíc existuje řada zařízení, která komunikují na více kanálech. Příkladem takového zařízení s více kanály může být multimediální klávesnice, která využívá druhý kanál pro odesílání informací o stisku multimediálních kláves. Jedna z těchto klávesnic byla odzkoušena.

4 Demonstrační aplikace pro klávesnici

Co je to klávesnice zmiňuje kapitola 1.4.1. Tato kapitola pojednává o funkci *keyboardLoop()*. O tom jak vznikala a co je zapotřebí si uvědomit než se začalo se zpracováváním výstupu zaslaného klávesnicí.

4.1 HID Usage Tables

Dokument rozšiřující specifikaci HID s názvem *HID Usage Tables* [9] je v současné době ve verzi 1.12. V něm je v kapitole pojednávající o klávesnici tabulka předdefinovaných hodnot, které zasílá klávesnice po stisku klávesy. Není nutné aby klávesnice všechny tyto tlačítka obsahovala, avšak pokud takové tlačítko má musí číselný kód, který navrací po svém stisku být shodný s hodnotou v tabulce.

Je zde předdefinováno 256 hodnot. Z tohoto množství je 13 hodnot rezervováno pro výrobce zařízení. Ti si mohou na tento kód připojit vlastní reakci napsáním speciálních ovladačů pro klávesnici. Protože klávesnice nemívají tolik kláves jsou zde zvláštní hodnoty pro rozšíření multimediálních a jiných klávesnic. Příkladem by mohly být kódy pro tlačítko „Kopírovat“, „Vložit“, „Zesílit zvuk“ nebo „Přehrávat“. Specifikace počítá i s národnostními zvláštnostmi, takže zde jsou vyhrazené kódy pro několik kláves se symboly zvoleného jazyka. První čtyři hodnoty tabulky odpovídají chybovým hlášením zasílaným směrem od klávesnice k systému.

Tabulka sepsaná v tomto dokumentu počítá s dvouúrovňovou strukturou. Pro přístup do druhé úrovně je třeba stisknout klávesu Shift nebo aktivovat klávesu Caps Lock. Ve druhé úrovni se nachází velká písmena a nad národními symboly jsou umístěny číslice.

Třetí úroveň se aktivuje stiskem klávesy Alt Gr a česká norma s ní ve výchozím stavu nepočítá. Nachází se zde speciální symboly vyskytující se ve znakové sadě systému. Česká norma ČSN 36 9050 počítá s dvouúrovňovou strukturou[1].

Tabulka z tohoto dokumentu je použita v programu jako pole struktur s názvem **keyboard**. Každá položka pole nese v sobě informaci o třech úrovních každé z kláves a krátký komentář převzatý ze specifikace.

Protože jde pouze o demonstrační aplikaci jsou zde zahrnuty pouze tisknutelné znaky a ostatní jsou nahrazeny hodnotou 0. Tak aby nedocházelo k výpisu matoucích symbolů. V případě potřeby obsluhy i těchto kláves je možno je v budoucnu nahradit funkcí pro jejich obsluhu. Problém obecného řešení spočívá v tom, že není jasné k čemu bude připojen modul VDIP1. Jiný druh obsluhy šipek na

1 Čtvrtá úroveň se zadává kombinací Alt Gr + Shift + klávesa

klávesnici si žádá posuv kurzoru na LCD displeji FITkitu a posuv textového kurzoru na monitoru připojeném přes VGA rozhraní.

4.1.1 Znaková sada ISO/IEC 8859-2

Pro správnost zobrazovaných symbolů bylo bezpodmínečně nutné určit znakovou sadu, kterou by podporoval LCD displej, zobrazovací terminál i program pro obsluhu výpisu na VGA výstup. Nakonec byla vybrána sada ISO 8859-2 spadající mezi sady pro středoevropské jazyky. Sada UTF-8 není podporována displejem, protože má příliš dlouhou informaci o symbolu. Oproti tomu sada cp1250 být sice zvolena mohla, avšak nešlo využít předem připravených knihoven s českými znaky pro FITkit.

Jelikož je zdrojový kód psán v UTF-8 bylo nutné psát hodnoty symbolů v číselné formě. Tedy alespoň symboly, jejichž číselná hodnota se pohybuje v horní polovině bytu.

Zvláštností je že znaková sada ISO 8859-2 v sobě nemá symbol pro měnu euro ačkoli se ve střední Evropě k platbě běžně používá.

4.2 Hlavní funkce obsluhy klávesnice

Pro korektní fungování funkce *keyboardLoop()* je nezbytně nutné mít připojené zařízení a nastavený komunikační kanál funkcí *sc()*. Za těchto podmínek se dá spustit smyčka vykonávající obsluhu. Před spuštění smyčky je vytvořeno několik proměnných, tak aby se dalo sledovat více funkcí klávesnice. Jsou zde proměnné pro detekci stisku více kláves, stisku znaménka nebo speciálních kláves. O všech těchto proměnných, jejich významu a obsluze se budou věnovat samostatné kapitoly.

Jak již bylo předem zmíněno program je vlastně smyčkou, ve které dochází k opakovanému čtení výstupních informací od klávesnice. Kanál sestavený funkcí *sc()*, je potřeba aby se dala využít funkce monitorovacího režimu. Tato funkce nastaví kanál pro čtení realizované příkazem DRD obvodu Vinculum.

Informace zachycené tímto způsobem jsou uloženy v globálním bufferu a jsou uvnitř jednotlivých funkcí interpretovány. Hlavní smyčka pouze filtruje neužitečné odpovědi. Při častých dotazech nebo nestisknuté žádné klávese přichází odpovědi, které nejsou užitečné pro další zpracování.

Poté dochází k volání funkcí zpracovávajících speciální klávesy, klávesy Caps Lock a Num Lock a znaménka. Po těchto akcích dochází k uložení tisknutelného znaku odpovídajícího kódování ISO 885-2 do proměnné „symbol“. Tato proměnná je posléze předána funkcím pro výpis symbolu v terminálu, LCD displej nebo na monitor připojený přes VGA výstup modulu FITkit.

4.2.1 Zpráva od klávesnice

Správně zachycená zpráva odeslaná klávesnicí se skládá z osmi byte informací a ukončovacího znaku prompt.

Příklad hlavní části dat zaslaných klávesnicí (hexadecimální reprezentace):

20 00 10 00 00 00 00 00

20 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00

První řádek příkladu reprezentuje současné stisknutí kláves Shift a M, tedy výpis malého písmene m.. Druhý řádek ukazuje odpověď na puštění klávesy M a třetí ukazuje puštění poslední stisknuté klávesy.

V pořadí první byte v sobě skrývá informaci o speciálních klávesách (anglicky Modifier key). Druhý byte je rezervován pro možná budoucí rozšíření specifikace. Následujících šest hodnot reprezentuje hodnoty stisknutých kláves. Hodnoty dalších byte se zobrazí při současném stisku více kláves. Proto je teoreticky možno pracovat s klávesovou zkratkou o šesti klávesách. V praxi je tento počet kláves dostačující. První byte v sobě také dokáže kódovat vícenásobný stisk.

Pro úplnost je třeba dodat, že při uvolnění klávesy dojde k odeslání zprávy s vynulovanou hodnotou po původním stisku.

4.2.2 Speciální klávesy (Modifier keys)

Tyto klávesy mění v kombinaci s jinou klávesou její původní význam. Obvykle samostatný stisk těchto kláves nevyvolá žádnou reakci a nevypíše tisknutelný znak. Funkce *KeyIdent()* se stará o správnou identifikaci stisku těchto kláves a mění hodnotu informační proměnné zodpovědné za danou klávesu.

Pořadí bitu	Klávesa	Proměnná
0	Levý Ctrl	fctrl
1	Levý Shift – druhá úroveň	fshift
2	Levý Alt (Alternative)	falt
3	Levá GUI klávesa (klávesa Windows)	fwin
4	Pravý Ctrl	fctrl
5	Pravý shift – druhá úroveň	fshift
6	Alt Gr (Alternative Graphic) – třetí úroveň	faltgr
7	Pravý GUI (meta, Fn, command)	fwin

Tabulka 4.1 – Speciální klávesy a jejich identifikační bity

4.2.3 Caps Lock

Klávesa Caps Lock slouží pro permanentní přepnutí do druhé úrovně tisknutelných znaků. Obsluha stisku této klávesy je prováděna ve funkci KeyIdent(). Protože si klávesnice sama nepamatuje stisk této klávesy je potřeba pro to využít paměť FITkitu, Proměnná CapsLock je zkontrolována vždy při stisku nové klávesy. A dále je při kladné hodnotě zpracována tak, že se nastavý proměná fshift. Dojde tedy vždy k nastavení stisku klávesy Shift. Klávesa se tedy chová přesně stejně jako pod operačním systémem.

V současné době není zpracována obsluha LED diod v klávesnici, tak aby jejich stav odpovídal stiskům kláves. Více o tomto tématu v kapitole 6.

4.2.4 Num Lock

Touto klávesou lze deaktivovat numerickou část klávesnice (anglicky Numpad). Situace je zde vpodstatě stejná jako u klávesy Caps Lock s tím rozdílem, že nedochází k změně signalizaní proměnné, ale původní hodnota reprezentující číslo je nahrazena nulou.

Opět lze prohlásit že chování klávesy je identické s chováním pod operačním systémem.

4.2.5 Znaménka

Čeština využívá tří znamének nad písmeny – čárka, háček a kroužek. Všechny tyto symboly se mohou objevit nad velkými i malými písmeny.

Jsou dva způsoby jak napsat písmeno sjedním z těchto znaků. Prvním je využití kláves s předem vyhrazeným prostorem pro národnostní znaky. Na těchto klávesách však nikde není velké písmeno v kombinaci s požadovaným znakem. Proto je zde druhý způsob kombinující postupný stisk dvou kláves. Operační systém se tak postará o uložení informace, že byl stisknut znak znaménk a následně po stisku písmene dojde k vypsání písmene v kombinaci se znaménkem. Tímto způsobem lze psát velké i malé znaky se znaménky.

První způsob psaní přímeho znaku se znaménkem funguje bez problémů. Ale v druhém případě již není program funkční. Bohužel nelze znaky se znaménky a bez znamének vytvořit jednoduchým přičtením nebo odečtením konstanty, tak jako je tomu u velkých a malých písmen. Bylo by potřeba vytvořit pro každé znaménko vytvořit jakousi tabulku. V ní by byly velká i malá písmena se znaménkem zbytek prázdné pozice. Takový postub by šlo uplatnit i na další zanménka, která se v češtině nevyskytují. Mezi tato diakritická znaménka patří například přehláska, stříška, cedilla, tylda, tečka nebo ogonek. Znamének je asi 18 a pro každe by tak musela existovat tabulka 2x26 char. To je téměř 1 KB paměti.

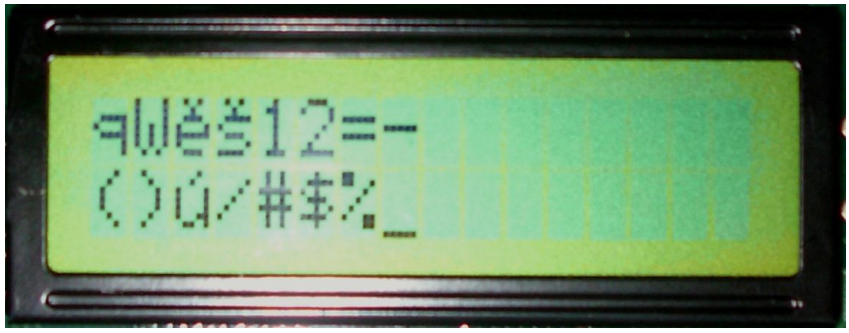
4.2.6 Stisk více kláves najednou

Tato problematika je částečně popsána v kapitole 4.2.1 Zpráva od klávesnice, kde je popsána forma příchozích zpráv. Protože smyslem této práce nebylo plně implementovat funkci klávesnice je v aplikaci při prozkoumání zdrojového kódu jen naznačena cesta jak postupovat při detekci vícenásobného stisku. Jde pouze o nastavení proměnné NextButton na hodnotu druhé stisknuté klávesy. Program tuto hodnotu dál nijak nevyužívá.

4.3 LCD – výpis symbolů na displej

Pro správnou funkci výpisu českých znaků na dislpleji LCD je v programu použita knihovna *display_cz.h*, která se stará o definici a výpis znaků horní poloviny bytu. Program posléze již jen voláním funkce *lcd_char()* s parametrem proměnné symbol realizuje výpis znaku a inkrementaci textového kurzoru na displej.

Bohužel není kompletně sestavena tabulka simbolů pro znakovou sadu, proto se nemusí některé znaky zobrazit správně. Navíc se zdá že některé symboly v klasické ASCII tabulce displeje neodpovýdají těm, které by se měly zobrazovat. Patrné je to u znaku "{" a "}".



Obrázek 4.1: Ukázka zapsaného textu na displeji FITkitu

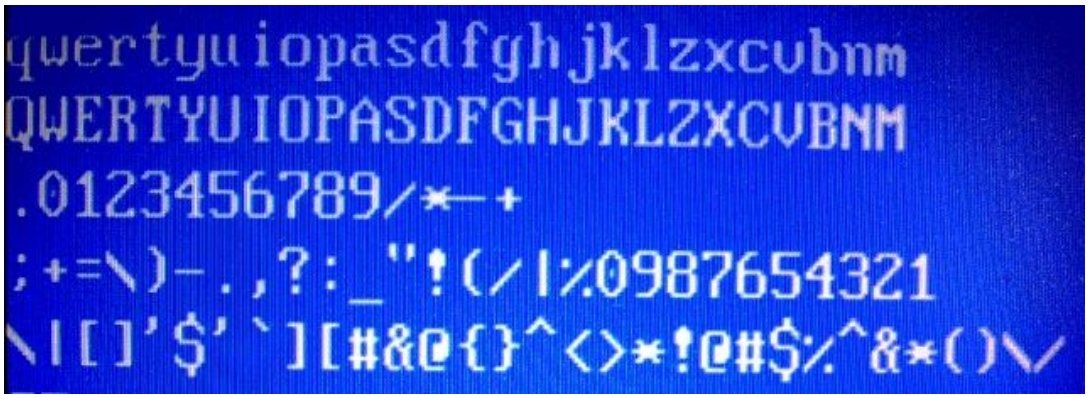
4.4 VGA – propojení s externím monitorem

Problematika psaní přímo na připojený monitor je poněkud složitější než u LCD. Úspěšně se podařilo klávesnici skrze modul VDIP1 a FITkit připojit k monitoru a zapisovat text přímo z klávesnice.

Druhá aplikace, která byla vytvořena realizuje grafický terminál. Pro hladký chod bylo nutné přidat knihovnu *vga_text.h* zařizující programovou obsluhu portu VGA. Navíc dodat program pro FPGA s názvem *top_level.vhd* a font nahrávaný vždy při spuštění.

Bohužel oproti ostatním výpisům v terminálu nebo na displeji je třeba se spokojit pouze s výpisem symbolů ASCII. Proto aby nedocházelo k zobrazování nesmyslných a matoucích hodnot je ve funkci *keyboardLoop()* vložena podmínka, která nastavuje symboly s numerickou hodnotou větší jak 127 na 0 a až po této akci jsou odeslány k výpisu na obrazovku.

Ve výchozím stavu je výpis na port VGA zakomentován v celém programu i souboru Makefile, ale prostým odstraněním komentářů a překladem zdrojových kódů ho lze uvést opět do chodu.



Obrázek 4.2: Ukázka zapsaného textu na monitoru připojeném k FITkitu

5 Demonstrační aplikace pro myš

Definicí tohoto polohovacího zařízení se zabývá kapitola 1.4.2. V této části dokumentu je popsáno jak je řešena obsluha , komunikační protokol a způsob reprezentace zaslaných signálů.

5.1 Zpráva od myši

Protokol definovaný ve specifikaci HID [5] je vytvořen s poměrně velkou benevolencí pro možná rozšíření. Tabulka 5.1 zobrazuje povinné části protokolu. Od třetího byte muže obsah výrobce libovolně definovat dle svých požadavků.

Byte	Bity	Popis
0	0	Tlačítko jedna
	1	Tlačítko dvě
	2	Tlačítko tři
	4 - 7	Ponecháno na výrobci
1	0 - 7	Změna souřadnic na ose X
2	0 - 7	Změna souřadnic na ose Y
3 - n	0 - 7	Ponecháno na výrobci (volitelné)

Tabulka 5.1: Popis bytů v příchozí zprávě od myši, převzato z [9]

Z tabulky je patrné, že se zde nepočítá s kolečkem. To je právě příklad možného rozšíření, které obecně převzali výrobci a informace o pohybu kolečka se tak nachází zpravidla na 3 bytu. Podrobnou reprezentací hodnot se zabývají ty části textu, které se vztahují k práci s reprezentací dané informace. Jak již bylo uvedeno v příkladu kapitoly 1.4 existuje celá řada možných rozšíření.

Myš využívaná pro vývoj aplikace měla zprávu délky 4 byty.

5.2 Hlavní funkce obsluhy myši

Princip činnosti je identický s klávesnicí. V hlavním programu je opakovaně volána funkce obvodu DRD, která zachytává zprávy připojeného zařízení. Následně jsou odfiltrovány neužitečné informace (selhání příkazu, prázdná odpověď). Poté již je byte po bytu zpracována zpráva samostatnými funkcemi, které se starají o rutiny okolo interpretace hodnot. Hlavní funkce se jmenuje *mouseLoop()*.

5.2.1 Tlačítka

ButtonIdent() vyhodnocuje stisk tří základních tlačítek a provádí grafický výstup stisků na LCD.

Funkce vyžaduje pouze byte s pořadovým číslem 0, jehož interpretace se řídí tabulkou 5.1. Provádí tedy pomocí bitových operací identifikaci stisknutých tlačítek. Při detekci stisknutého tlačítka je vykreslen obdélníček na displeji. Pravé a levé tlačítko má svůj vyhrazený obdélník odpovídající poloze na myši. Prostřední tlačítko je kombinované s kolečkem, a proto je reprezentováno rozsvícením obdélníčků odpovídajících i pohybu kolečka.

Obrázky pro lepší pochopení činnosti jsou uvedeny v kapitole 5.2.4.

5.2.2 Posun v prostoru

Funkce *XxY()* se stará o výpis rozdílových souřadnic na displej FITkitu a potřebuje jako vstup rozdílové souřadnice obou os, tedy hodnoty signed char.

Myš poskytuje informaci o své pozici pomocí rozdílových souřadnic. Neustále snímá a v reálném čase porovnává snímaný povrch na kterém se pohybuje a vyhodnocuje posun. Tento posun posléze vyjádří pomocí dvou znaménkových čísel v rozsahu -127 až 127. První byte reprezentuje změnu na x-ové ose a druhý na ose Y.

Při předpokladu že výchozí bod obrazovky je v levém horním rohu stačí operačnímu systému hodnoty obdržené od myši pouze přičíst. Souřadnicový systém polohovacího zařízení je převrácený oproti systému.

Hodnota obou souřadnic je vypsána na prvním řádku displeje včetně znaménka, které k souřadnici patří. Hodnoty jsou odděleny symbolem znakem „x“. Obrázky v sekci 5.2.4 ukazují možnosti zobrazení souřadnic při pohybu do různých sektorů.

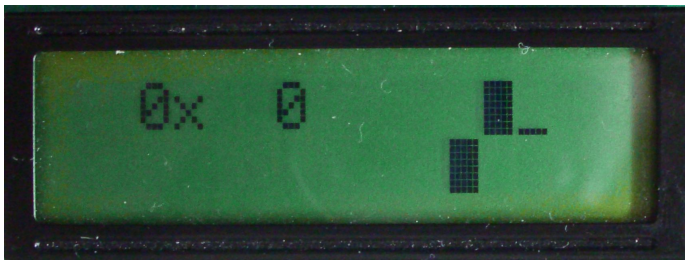
5.2.3 Kolečko

Pro zpracování této informace je použit grafický výstup v podobě dvou nad sebou stojících obdélníčků. Při pohybu kolečka se chovají tyto obdélníčky tak, že se vždy rozsvítí ten jehož směrem by se pohyboval posuvník na okraji okenní aplikace na počítači.

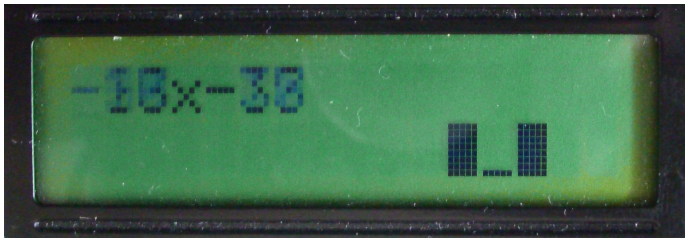
Funkce provádějící tuto činnost se jmenuje *scroll()* a postačuje jí poslední 3 byte zaslaný zařízením. Opět se jedná o znaménkovou hodnotu reprezentující relativní pohyb kolečka. I při relativně rychlém otáčení docházelo k odesílání hodnot v rozsahu <-5;5>.

5.2.4 Fotky ukazující grafický výstup pro myš

Obrázek 5.1: Rozdílové souřadnice posledního pohybu



Obrázek 5.2: Stisk levého tlačítka myši a zároveň otočení kolečkem směrem k sobě



Obrázek 5.3: Pohyb myši a současné stisknutí pravého a levého tlačítka

Obrázek 5.4: Pohyb myši a současné držení všech tří tlačítek

5.3 Ladění

Pro zachycení komunikace na nejnižší úrovni USB byly využity programy USBTrace [13] a USBSnoop [14]. Výstup prvního jmenovaného je přidán jako příloha. Dokument HTML v sobě nese zachycené pakety, které si vyměnila myš při připojení k počítači.

Důležité jsou především pakety URB_FUNCTION_CONTROL_TRANSFER, které v sobě nesou SetupPacket informace. Při práci se zařízením, které Vinculum nepodporuje lze studiem těchto paketů nastudovat příkazy připojeného zařízení současně.

6 Možnosti rozšíření práce

Kapitola se snaží nastínit možnosti dalšího vývoje okolo propojení těchto dvou modulů. Proto se pokusí upozornit na doposud nevyřešené problémy, návrhy rozšíření, dokončení a praktické návrhy.

Jedním z návrhů na zlepšení je dokončení tabulky symbolů ISO 8859-2. V současné době je pro výpis znaků na LCD použita ASCII tabulka + jakési rozšíření s většinou znaků používaných v českém textu umístěných na pozicích odpovídajících kódování ISO. Avšak na klávesnici se vyskytuje i řada znaků, které nejsou v současné době v tomto rozšíření zapsaná. Konkrétně je to například těchto pár znaků „§ \n ´ \ ° ˇ | \t ~ } { Ł“ jejichž číselná hodnota se na displeji zobrazí jako zcela jiný symbol. Proto by bylo dobré dodefinovat i zbylé znaky z horní poloviny bytu, tak aby bylo kódování kompletní. Jde o manuální práci v podobě definování nových znaků do knihovny *display_cz.h*.[2]

O problematice znamének se již zmiňuje kapitola 4.2.5. Detekce více kláves je sice zprovozněna avšak aplikace s touto informací nijak nepracuje. Praktické využití by se mohlo naskytnout například při stisku kláves pro ukončení nějaké navržené aplikace.

Doposud nevyřešený problém je signalizace LED diod indikujících zmáčknutý Caps Lock a Num Lock. Řešení tohoto problému je teoreticky proveditelné skrze příkaz obvodu Vinculum DSD, kterým zašleme požadavek „Set_Report(Output) request“ podle specifikace USB.

Zajímavým rozšířením by mohlo být spouštění obslužných programů *keyboardLoop()* a *mouseLoop()* jako samostatná vlákna s využitím např. realtime OS. Pokud by se vyvedl na nepájivé pole i druhý USB port mohlo by dojít o obsluze obou připojených zařízení.

2 Příklad jednoho z chybějících znaků:

Tcharmatrix _lcd_cz_L_ = {0x08, 0x08, 0x0A, 0x0C, 0x08, 0x18, 0x08, 0x0F}; //Ł

7 Závěr

Využití propojení USB HID zařízení a platformy FITkit se jeví jako zajímavá cesta k budoucím možnostem platformy. USB porty jsou skutečně maximálně universálním rozhraním a stávají se nepostradatelnou součástí našich životů, proto je pro studenty prakticky nutné získat zkušenosti s vývojem pro toto rozhraní již na škole. Bohužel tohoto cíle se s obvodem Vinculum nedosáhne v potřebné míře.

Zařízení typu HID přináší taktéž velký potenciál v podobě větší spolupráce programů s člověkem. Doposud jediným způsobem řízení změny v běžícím programu na FITkitu byla klávesnice s šestnácti tlačítky, nově je reálné pro interakci s právě prováděným programem využití plnohodnotné klávesnice nebo myši. Tento dokument vysvětluje práci při zpracování signálů pouze dvou nejrozšířenějších HID zařízení, avšak obdobným postupem lze proniknout do problematiky práce s joystickem nebo jiným zařízením. Práce mimo jiné prokázala nespornou výhodou použití obvodu Vinculum kterou je skutečnost, že díky obsaženému firmware není nezbytně nutné, aby programátor studoval detaily USB protokolu. Tento obvod poskytuje jakousi základní abstrakci nad operacemi.

Projekt je v současné době úspěšně provázaný s existujícími projekty okolo platformy FITkit. Využívá funkčního VGA rozhraní a české lokalizace pro LCD. Existující rozhraní IDE pro FITkit dokáže připojit několika gigabytový disk. Stejnou funkcionalitu zajistí modul VDIP1 a externí disk do USB.

Práce měla přínos především pro projekt FITkitu, kde se podařilo vytvořit knihovní funkce pro práci s dalším zařízením schopným komunikovat s tímto modulem. Osobní přínos této práce spatřuji v bližším pochopení významu standardizace hardwarových a softwarových výrobků, proniknutí do struktury USB a třídní hierarchie komunikačních protokolů této sběrnice. Zajímavou zkušeností je propojení těchto jinak samostatně fungujících komponent v kompaktní pracující celek.

Literatura

[1] Stránky projektu FITkit. [online], poslední aktualizace 7. 5. 2010, [cit. 2010-05-07].
URL <http://merlin.fit.vutbr.cz/FITkit/>

[2] FTDI: VDIP1 Vinculum VNC1L Module Datasheet . [online], Verze 1.0,
©2009 Future Technology Devices International Limited, poslední aktualizace 10. 12. 2009, [cit. 2010-05-07].
URL <http://www.vinculum.com/documents/datasheets/DS_VDIP1.pdf>

[3] FTDI: Vinculum Firmware User Manual . [online], Verze 2.05,
©2009 Future Technology Devices International Limited, poslední aktualizace 8. 4. 2008, [cit. 2010-05-07].
URL <http://www.vinculum.com/documents/fwspecs/UM_VinculumFirmware_V205.pdf>

[4] Pandatron.cz: Vinculum - seriál. [online], poslední aktualizace 10. 7. 2007, [cit. 2010-05-07].
URL <http://pandatron.cz/?158&vinculum_dil_1>

[5] Device Class Definition for HID 1.11. [online], poslední aktualizace 7. 5. 2010,
[cit. 2010-05-07].
URL <http://www.usb.org/developers/devclass_docs/HID1_11.pdf>

[6] hw.cz: USB 2.0 - díl 1. [online], poslední aktualizace 22. 2. 2005, [cit. 2010-05-07].
URL <http://hw.cz/Rozhrani/ART1232-USB-2.0---dil-1.html>

[7] Wikipedia: Počítačová myš. [online], poslední aktualizace 8. 2. 2010, [cit. 2010-05-07].
URL <http://cs.wikipedia.org/wiki/Počítačová_myš>

[8] Schéma zapojení FITkit verze 2.0. [online], poslední aktualizace 7. 5. 2010, [cit. 2010-05-07].
URL <http://merlin.fit.vutbr.cz/FITkit/download/schematic_v20.pdf>

[9] HID Usage Tables 1.12. [online], poslední aktualizace 7. 5. 2010, [cit. 2010-05-07].
URL <http://www.usb.org/developers/devclass_docs/Hut1_12.pdf>

[10] FTDI: VDIP1 Schematic Printout. [online],
©2009 Future Technology Devices International Limited, [cit. 2010-05-07].
URL <http://www.vinculum.com/documents/schematics/VDIP1%20Schematic%20Prints.pdf>

[11] AXELSON Jan. *USB complete: everything you need to develop custom USB peripherals.* New York: Lakeview Research. 2005. ISBN 1-931448-02-7.

[12] Stránky výrobce k obvodům rodiny Vinculum.
©2009 Future Technology Devices International Limited,
URL <http://www.vinculum.com>

[13] Stránky programu USBTrace. URL <http://www.sysnucleus.com/>

[14] Stránky programu Wireshark. URL <http://www.wireshark.org/>

Seznam příloh

Příloha A - mouse.html – výstup aplikace USB Trace, zachycená komunikace mezi myší a počítačem

Příloha B - Zdrojové texty

Příloha C - CD obsahující zdrojové kódy a programovou dokumentaci